(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 August 2001 (16.08.2001)

PCT

(10) International Publication Number
WO 01/59987 A2

(51) International Patent Classification[7]: H04L 12/00

(21) International Application Number: PCT/US00/41161

(22) International Filing Date: 12 October 2000 (12.10.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/500,404 8 February 2000 (08.02.2000) US

(71) Applicant *(for all designated States except US)*: **SYNDEO CORPORATION** [US/US]; Suite 120, 20195 Stevens Creek Blvd., Cupertino, CA 95014 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **GRIGGS, Theodore, Leon** [US/US]; 85 Ranch Road, Woodside, CA 94062 (US).

(74) Agents: **MICHAEL, J., Mallie** et al.; Blakely, Sokoloff, Taylor & Zafman LLP, 7th Floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/59987 A2

(54) Title: METHOD AND APPARATUS FOR DIVERTING A PACKET-SWITCHED SESSION UTILIZING AN INTELLIGENT PROXY AGENT

(57) Abstract: A method of diverting a packet-switched session commences with the reception, at a proxy agent, of a session invitation request propagated from a caller to request establishment of a session with a callee. The proxy agent detects a divert trigger event. Responsive to the divert trigger event, the proxy agent originates a fake session initiation request, the fake session initiation request being constructed to identify the caller as the originator thereof. The fake session initiation request is communicated to a third party to establish the session between the caller and a third party. The proxy agent may optionally establish a session between the caller and the callee responsive to the session invitation request, prior to communicating the fake session initiation request to the third party. If so, the proxy agent, responsive.

METHOD AND APPARATUS FOR DIVERTING A PACKET-SWITCHED SESSION UTILIZING AN INTELLIGENT PROXY AGENT

FIELD OF THE INVENTION

The present invention relates generally to the field of packet-switched networking and, more specifically, to a method and apparatus of switching a session conducted over a packet-switched network between different callees utilizing, for example, an intelligent proxy agent or server.

BACKGROUND OF THE INVENTION

As a voice and data networks converge, new protocols are being standardized to promote interoperability among such voice and data networks. One such protocol is the Session Initiation Protocol (SIP) that handles the signaling portion of a session, for example a call. The Session Initiation Protocol is an application-layer control, or signaling, protocol for creating modifying and terminating sessions among two or more participants. Such sessions may include multi-media conferences, calls or multi-media distribution channels. SIP may invite both persons or automated functions (e.g., a media storage service) to participate in a session.

SIP provides a number of message types that may be utilized to handle the signaling or control portion of a session. These messages include an invitation message (INVITE), an acknowledge message (ACK) and a terminate message (BYE). Two of the primary purposes of SIP are the initiation of a session utilizing the INVITE message and the tearing down of a session utilizing the BYE message.

SIP labels a client application that initiates or responds to a SIP request as a User Agent Client (UAC), or a call user agent. A server application that contacts a user when a SIP request is received is labeled a User Agent Server (UAS). The UAS further returns a response to the user to the originator of the SIP request. It should be noted that a particular application might be capable of functioning both as a client application and a server application.

A UAS may, in respective proxy and redirect modes, function as a proxy server or a redirect server or agent. In a proxy mode, a UAS may function as both a server and a client for the purposes of making requests on behalf of other clients. Requests are serviced either internally or by passing requests on, possibly after translation, to a further server or client. In proxy mode, a UAS interprets and possibly re-writes a request message before forwarding it. In redirect mode, a UAS may accept a SIP request, map a callee address to a new address, and return that new address to the client that originated the SIP request.

Figure 1A is a protocol diagram illustrating a prior art sequence 10 of message communications according to SIP. A first facility 12 includes a client application 14 (e.g., an Internet client application) that operates as a UAC and a server application 16 that function as a UAS for the first facility 12. The first facility 12 is coupled via a network 18, in exemplary form of the Internet, to a second facility 20 that includes a further server application 22 that operates as a UAS, a location server 24 and a further client application 26 that operates as a UAC. In the exemplary sequence 10, the server application 22 of the second facility 20 is operating in proxy mode.

The sequence 10 commences with the origination at, and communication from, the client application 14 of an INVITE message 28, directed to the further client application 26, for the establishment of a session (e.g., an Internet telephone call) between the client applications 14 and 26. To this end, the INVITE message 28 has a header that includes a TO field that specifies a network address of the client application 26 (e.g., the callee) and a FROM field that specifies a network address of the client application 14 (e.g., the caller). For example, the network address may employ DNS-style addressing that specifies a server domain or host at which an address resides (e.g., SIP: user@domain or SIP: user@host). The INVITE message 28 is communicated via the server application 16 and the network 18 to the server application 22 that, in proxy mode, issues a LOOKUP message (e.g., LOOKUP {user}) to the location server 24. The location server 24 resolves the address, and issues a RETURN message

32 to the server application 22 specifying an address at which the client application 26 may be contacted. The server application 22 then forwards the INVITE message 28, possibly modified to include an alternative network address, to the client application 26.

Responsive to the INVITE message 28, the client application 26 issues an OK message 34, via the server applications 22 and 16 and the network 18, to the client application 14. The OK message 34 provides an indication to the client application 14 that the client application 26 has agreed to participate in a session, and a message body of the OK message 34 may indicate the capabilities of the client application 26.

Responsive to the OK message 34, the client application 14 then issues an ACKNOWLEDGE message 36 to the client application 26 via the server application 16, the network 18 and the server application 22. The ACKNOWLEDGE message 36 confirms that the client application 14 has received a final response to the initial INVITE message 28.

For each of the entities (e.g., a UAC or a UAS) that any one of the above message traverses on a network path between into points of a session, a VIA field may be added to the general header of the relevant message. Accordingly, the VIA fields included within the general header of a message provide an indication of the network path taken by the relevant message.

The purpose of the INVITE message 28 service is thus to establish a session between, for example, a caller and a callee. Either party to a session may then terminate, or tear down, the session by issuing a BYE message (not shown) to the other party.

Figure 1B is a block diagram providing an alternative representation of the prior art sequence 10 of message communications described above with reference to **Figure 1A**, and shows further details regarding the content of headers (e.g., the VIA fields) of the various messages that are communicated between the various entities along the network path.

SIP supports user mobility by facilitating the redirecting of requests to a user's current location. A user may register a current location with a location

server 24, in which case SIP messages may be redirected to the registered location. **Figure 2** is a protocol diagram illustrating a further prior art sequence 38 of message communications according to SIP, where a redirect function is performed by a UAS operating in a redirect mode. As described above with respect to **Figure 1**, the client application 14 issues the INVITE message 28 to the server application 22, that in turn issues a LOOKUP message 30 to the location server 24. In this case, however, the RETURN message 32 indicates an alternative address (e.g., an alternative domain or host) that has been registered as the current location with the location server 24 for the client application 26, or the user associated with the client application 26. In this case, the server application 22, operating in the redirect mode, issues a MOVED TEMPORARILY message 40 to the client application 14, the message 40 specifying the current location (e.g., a redirect address) of the client application 26 (e.g., the callee). The client application 14 then issues an ACKNOWLEDGE message 42 to the server application 22 and a fresh INVITE message 44 to the redirect address. A server application 27 at a further facility 29 may then perform a lookup operation, as described with respect to **Figure 1**, and forward the INVITE message 44 to the client application 26.

SUMMARY OF THE INVENTION

A method of diverting a packet-switched session includes receiving a session invitation request propagated from a first party to request establishment of a session with a second party. A divert trigger event is detected. Responsive to the divert trigger event, a fake session initiation request is originated at an intermediate party located intermediate the first and second parties on a network path, the fake session initiation request being constructed to identify the first party as the originator thereof. The fake session initiation request is communicated to a third party to establish the session between the first and third parties.

Other features of the present invention will be apparent from the accompanying drawings and from the detailed description that follows.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements and in which:

Figure 1A is a protocol diagram illustrating a prior art sequence of message communications according to the Session Initiation Protocol (SIP).

Figure 1B is a block diagram providing an alternative representation of the prior art sequence of message communications illustrated in **Figure 1A**.

Figure 2 is a protocol diagram illustrating a further prior art sequence of message communications according to SIP.

Figure 3 is a flow chart illustrating an exemplary method, according to one embodiment of the present invention, of diverting a packet-switched session from a first callee to a second callee.

Figure 4 is a flow chart illustrating an exemplary method, according to one embodiment of the present invention, of reverting a packet-switched session from a second callee back to a first callee.

Figures 5A and **5B** are block diagrams illustrating exemplary sequences, according to respective embodiments of the present invention, of message communications, whereby a packet-switched session may be diverted from a first callee to a second callee.

Figure 6 is a block diagram illustrating an exemplary embodiment of a server application and a client application deployed within, for example, an inbound call center.

Figure 7 illustrates an exemplary communication of messages between a client application and a server application.

Figure 8 is a block diagram illustrating an exemplary intermediate party, according to one embodiment of the present invention, in the form of a server application which is shown to include a detector.

Figure 9 is a state diagram illustrating a number of exemplary states implemented by a "queue and divert" state machine, according to one embodiment of the present invention.

Figure 10 is a state diagram illustrating a number of exemplary states implemented by a "de-queue and ring" state machine, according to one embodiment of the present invention.

Figure 11 is a diagrammatic representation of a machine in the exemplary form of a computer system within which a set of instructions may be executed.

DETAILED DESCRIPTION

A method and apparatus for diverting a packet-switched session are described. In the following description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be evident, however, to one skilled in the art that the present invention may be practiced without these specific details.

For the purposes of the present specification, the term "party" shall be taken to include any entity that participates in, or contributes towards, a session, and shall be taken to include, but not limited to, an Internet telephony application, a multi-media conference application, and a multi-media distribution application. The term "party" shall also be taken to be inclusive of the terms "end-point", "caller" and "callee".

While an exemplary embodiment is described below with reference to the Session Initiation Protocol (SIP), it will be appreciated that the present invention is not limited to utilization within the context of SIP, and could be implemented within the context of any number of other protocols.

Figures 3 is a flowchart illustrating an exemplary method 50 of diverting a packet-switched session (e.g., an Internet Protocol (IP)) from, for example, a first callee to a second callee. The diverting of the session in the exemplary embodiment is performed by an intelligent proxy server acting as a UAS, and allows the relevant UAS to regain control of the session at any time and at the discretion of the UAS. The diversion of the session, and the ability to regain control of the session, may be useful in the number of applications. For example, where a proxy server is associated with an inbound call center, and an inbound call is required to be queued pending the availability of a live operator, the present invention facilitates the diversion of the call to, for example, an outside service that may provide a hold service (e.g., music and/or advertisements). The proxy server is then able to regain control of the session when a live operator becomes available, and establish the session between the caller and the live operator.

The method 50 will also be described with reference to **Figures 5A** and **5B**, which are block diagrams illustrating exemplary sequences 80 and 81 of message communications whereby a packet -switched session (e.g., an Internet Protocol (IP) session) may be diverted, for example, from a first callee to a second callee.

The method 50 commences at block 52 with the actual or attempted establishment of a packet-switch session between first and second parties via a proxy server, for example in the manner described above with reference to **Figures 1A** and **1B**. At block 54, a session divert trigger event is detected. The detected trigger event could comprise any number of events, depending on the context of the session. For example, where the client application 84 has issued an INVITE message (not shown) to an inbound call center, the trigger event may be the placement of a call (supported by the session between the client

application 84 and a client application 86) in a queue pending availability of the client application 86, or an agent utilizing the client application 86, to handle the call.

In an alternative embodiment, the trigger event may be a call transfer function implemented by the callee client application 86, for example, to transfer the call to a different client application 88. In this case, it would be desirable for a proxy server associated with the callee client application 86 to be able to regain control of the call in the event that the different client application 88 is not able to take the call or when the different client application ends a call.

In yet a further embodiment, the trigger event may be a call transfer function, or new call request, initiated by the caller client application 84 in an environment or application in which it is advantageous for a proxy server to maintain control of a session with the caller client application 84. For example, where the client application 84 is being utilized to make a number of consecutive Internet telephone calls via a calling card service (e.g., an AT&T calling card), a proxy server associated with the calling card service may wish to maintain a continuous session with the caller client application 84, but allow the session to be diverted to a number of callee client applications. The maintenance of such a continuous session is advantageous in that the user of the caller client application 84 is not be required to communicate monetary, account and/or identification information to the calling card service for each consecutive call supported by a continuous, single session. Numerous other applications of the described technology will be apparent to those skilled in the art, and trigger events for each of such applications could accordingly be detected at block 68.

In the event that a session has been established between first and second parties, such as for example the call 82 between the client applications 84 and 86 as illustrated in **Figure 5A**, the method 50 proceeds to perform the actions described at blocks 56, 58 and 59 in **Figure 3**. On the other hand, should a session not as yet have been established, and an attempt merely have been made to establish a session by, for example, the issuance of an INVITE message

93 from the caller client application 84, the actions at block 56, 58 and 59 will not be required.

The establishment of a session between the first and second parties (e.g., the client applications 84 and 88) prior to the session diversion is, it will be appreciated, not a requirement. **Figure 5B** illustrates the diversion of a session from a first callee to a second callee, where there is no pre-existing session between the caller and the first callee. Specifically, in this case, the call is diverted to the second callee without the establishment of any call, or session, with the first callee.

Figure 5B is a block diagram illustrating an exemplary sequence of message communications where a session is diverted to a second callee, without the prior establishment of a session with the first callee. In the exemplary embodiment, the caller client application 84 is shown to issue an INVITE message 93 addressed to the callee client application 86. The server application 90, as a proxy agent, absorbs this INVITE message, and issues a fake OK message 95 to the caller client application 84. The fake OK message 95 is constructed to include a network address identifying the second client application 86 (e.g., UAC_2) as the originating entity.

Responsive to the fake OK message 95, the first client application 84 issues n ACKNOWLEDGE message 97, addressed to the callee client application 86. The server application 90 again absorbs this message, and does not forward it to the callee application 86.

The exchange of messages 93, 95 and 97 accordingly leads the caller client application 84 to believe that a session has been established with the callee application 86, when in fact this is not the case. As will be discussed in further detail below, the server application 90 then proceeds to establish the diverted session with a further client application 88 (e.g., UAC_3) by issuing a fake INVITE message 100, absorbing an OK message 102 from the further callee application 88, and issuing a further fake ACKNOWLEDGE message 104 to the further callee application 88.

Returning to **Figure 3** and **Figure 5A**, at block 56, an intermediate party

located intermediate the first and second parties on a network path, generates and issues a fake session termination request to the second party that appears to have been generated upstream. Referring to the exemplary embodiment shown in **Figure 5A** where a pre-existing session exists between a caller and a callee, a server application 90, operating as a UAS in proxy mode (i.e., as a proxy agent), generates and issues a fake BYE message 92 to the client application 86. The BYE message 92 includes a header that falsely indicates the message 92 as having been originated at and issued from an upstream location and from the client application 84. The server application 90, as a proxy agent, further generates the fake BYE message 92 to include a number of VIA fields that falsely indicate the message 92 as having traversed a network path between the client application 84 and the server application 90. For example, the BYE message 92 may include a VIA field providing a fake indication that the message 92 traversed the caller server agent 94.

Examples of the values that may be attributed to fields within a header of a BYE message 92 are as follows:

BYE sip: uac_2@facilityB SIP/2.0
Via: SIP/2.0/UDP 100.100.100.2
Via: SIP/2.0/UDP 100.100.100.3
From:UAC_1 <uac_1@facilityA>
To: UAC_2 < uac_2@facilityB>
Call-ID: 187602141351@facilityA
Subject: Telephone Call

At block 58, the second party confirms receipt of the session termination request, and the intermediate party absorbs this confirmation so that is not communicated to the first party. For example, referring to **Figure 5**, the callee client application 86, responsive to the BYE message 92, issues an OK message 96 that is addressed to the caller client application 84. The server application 90 then absorbs, and does not forward, the OK message 96.

At block 59, the intermediate party issues a fake acknowledgement

message to the second party. For example, the server application 90 issues a fake ACKNOWLEDGE message 98 to the callee client application 86. The fake ACKNOWLEDGE message 98 also includes FROM and VIA fields indicating a fake originator and network path.

Accordingly, the actions taken at blocks 56, 58 and 59 cause the callee client application 86 to understand that the session (e.g., the call 82) has been terminated by the caller client application 84. The caller client application 84, on the other hand, has received no information that indicates to it that the status of the relevant session has changed.

At block 74, the intermediate party generates and issues a fake session initiation request to a third party. Referring to the exemplary embodiment shown in **Figure 5A**, the server application 90 generates and issues a fake INVITE message 100 to a further callee client application 88 (e.g., a hold service or alternative call recipient). As with the fake BYE message 92 described above, fake INVITE message 100 includes a header that falsely indicates the message 100 as having been originated at and issued from the client application 84. The server application 90, as a proxy agent, further generates the fake INVITE message 100 to include a number of VIA fields that falsely indicate the message 92 as having traversed a network path between the client application 84 and the server application 90. For example, the INVITE message 100 may include a VIA field providing a fake indication that the message 100 traversed the caller server agent 94.

Examples of the values that may be attributed to fields within a header of a BYE message 92 are as follows:

INVITE sip: uac_3@facilityC SIP/2.0
Via: SIP/2.0/UDP 100.100.100.2
Via: SIP/2.0/UDP 100.100.100.3
From:UAC_1 <uac_1@facilityA>
To: UAC_3 < uac_3@facilityC>
Call-ID: 187602141351@facilityA

Subject: Telephone Call

At block 76, the third party confirms receipt of the session initiation request, and the intermediate party absorbs this confirmation so that is not communicated to the first party. For example, referring to **Figure 5A**, the callee client application 88, responsive to the INVITE message 100, issues an OK message 102 that is addressed to the caller client application 84. The server application 90 then absorbs, and does not forward, the OK message 102.

At block 78, the intermediate party issues a fake acknowledgement message to the third party. For example, the server application 90 issues a fake ACKNOWLEDGE message 104 to the callee client application 88. The fake ACKNOWLEDGE message 104 also includes FROM and VIA fields indicating a fake originator and network path.

Figure 4 is a flowchart illustrating a method 64 of reverting a packet-switched session from, for example, a second callee to which a session was originally diverted back to a first callee to which the session was originally targeted by a caller. The method 64 commences at block 66 from a condition in which a session, for example supporting the call 106 shown in **Figure 5A**, has been established. At block 68, a session revert trigger event is detected. The session revert trigger event, as with the session divert trigger event, is application and context specific. **Figure 6** illustrates an exemplary embodiment where the server application 90 and the client application 86 are deployed in an inbound call center. In this example, the session revert event may be the communication of a BYE or a REGISTER message 110 from a client application 86 to the server application 90. The message 110 indicates to the server application 90 that the client application 86 has concluded a previous transaction or session , and is now available to handle the call 82 that was originally diverted to the client application 88. Responsive to the receipt of the message 110, the server application 90 may de-queue the call 82 and proceeded to establish the call between the client applications 84 and 86. In this example, the server applications 90 may employ a state machine that recognizes the

receipt of the message 110, or the de-queuing of the call 82, as a session revert trigger event.

At blocks 70,72, 73, 74, 76 and 78, the method 64 specifies a series of operations corresponding substantially to those described above with reference to blocks 56-63, except that the intermediate party (e.g., the server application 90) issues a fake session termination request (e.g., a fake BYE message 112) to the third party (e.g., the client application 88). Further, the intermediate party issues a fake session initiation request (e.g., a fake INVITE message 114) to the second party (e.g., the client application 86). The communication of these messages, the responses from the respective client applications 86 and 88, and the acknowledgement messages back to the client applications 86 and 88 from the server application 90 are illustrated in **Figure 7**.

Figure 8 is a block diagram illustrating an intermediate party in the exemplary form of an server application 90, which is shown to include a detector in the exemplary form of detect logic 120 that, as described above, detects both session divert and revert trigger events. The detect logic 120 is shown to have an awareness of the receipt of the BYE or REGISTER message 110, discussed above with reference to **Figure 6**, to enable the detect logic 120 to detect a session revert trigger event.

The detect logic 120 is also shown to monitor a call queue table 122, in which calls (e.g. , Internet telephony calls) are queued and de-queued to enable the queuing and de-queuing of calls to act as session divert and revert trigger events. It will be appreciated that, in other applications and contexts, the detect logic 120 may monitor other data structures specific to such further of applications that are suitable for signaling trigger event.

The server application 90 also includes a protocol module in the exemplary form of flow logic 124 that, responsive to the detection of a session divert or revert trigger event by the detect logic 92, accesses state tables 96 to perform the methodologies discussed above with reference to the flowcharts shown in **Figures 3** and **4**. The state tables 96 embodying two state machines, namely a "queue and divert" state machine 126 and a "de-queue and ring" state

machine 128.

Figure 9 is a state diagram illustrating a number of states implemented by the "queue and divert" state machine 126. A first "is available" state 130 evaluates, for example, whether a callee (e.g., the client application 86) is available to handle a call initiated by the issuance of an INVITE message from a caller (e.g., the client application 84). If so, the state machine 126 proceeds to a "ring" state 132, where the call is established with the callee. On the other hand, should the callee not be available to take the call, the state machine 126 proceeds from state 130 to a "queue" state 134, where the call is inserted into the queue table 122. In one embodiment, the server application 90 may maintain a set of queues in one or more queue tables 122. The call may be inserted into any queue in such a set of queues.

From state 130, the state machine 126 proceeds to an "invite" state 136, where an INVITE message is issued to an alternative callee (e.g., a hold service) in a manner described above. In a further embodiment, the state machine may include a "forward" state (not shown) where a received INVITE message is forwarded along to an original or another addressee (or callee) following establishment of a temporary call (or session) with the alternative callee.

Figure 10 is a state diagram illustrating a number of states implemented by the "de-queue and ring" state machine 128. A first "becomes available" state 140 monitors for the receipt at the server application 90 of a BYE message from a relevant client application. In the absence of any such BYE message, the state machine 128 oscillates between an idle state 142 and the "becomes available" state 140. On the detection of the issuance of a BYE message by a monitored client application, the state machine 128 progresses from state 140 to a "de-queue" state 144, where a relevant call is removed from the queue table 122. From state 144, the state machine 128 progresses to state 146, where the server application 90 proceeds to issue an INVITE message to the monitored client application to thereby establish the previously queued call between a callee and the caller (i.e., the monitor client application).

In summary, the present invention is advantageous in that it allows, for

example, a proxy agent to regain control of a call at anytime. A further benefit of the present invention is that may be utilized in a manner that is in compliance with SIP. Accordingly, a session may be diverted to an outside service (e.g., a hold service) provided by a third party that has no knowledge of the "fake" session initiation and termination messages. The present invention thus facilitates seamless interoperability with heterogeneous components.

Figure 11 shows a diagrammatic representation of machine in the exemplary form of a computer system 200 within which a set of instructions, for causing the machine to perform any one of the methodologies discussed above, may be executed. In alternative embodiments, the machine may comprise a network router, a network switch, a network bridge, Personal Digital Assistant (PDA), a cellular telephone, a web appliance or any machine capable of executing a sequence of instructions that specify actions to be taken by that machine.

The computer system 200 includes a processor 202, a main memory 204 and a static memory 205, which communicate with each other via a bus 206. The computer system 200 may further include a video display unit 208 (e.g., a liquid crystal display (LCD) or a cathode ray tube (CRT)). The computer system 200 also includes an alpha-numeric input device 210 (e.g. a keyboard), a cursor control device 212 (e.g. a mouse), a disk drive unit 214, a signal generation device 216 (e.g. a speaker) and a network interface device 218.

The disk drive unit 214 includes a machine-readable medium 215 on which is stored a set of instructions (i.e., software) 220 embodying any one, or all, of the methodologies described above. For example, the software 220 may comprise the detect logic 120, the flow logic 124, the state tables 96 and the queue table 122 described above with reference to **Figure 8**. The software 220 is also shown to reside, completely or at least partially, within the main memory 204 and/or within the processor 202. The software 220 may further be transmitted or received via the network interface device 218. For the purposes of this specification, the term " machine-readable medium" shall be taken to include any medium which is capable of storing or encoding a

sequence of instructions for execution by the machine and that cause the machine to perform any one of the methodologies of the present invention. The term "machine-readable medium" shall accordingly be taken to included, but not be limited to, solid-state memories, optical and magnetic disks, and carrier wave signals.

Thus, a method and apparatus for diverting a packet-switched session have been described. Although the present invention has been described with reference to specific exemplary embodiments, it will be evident that various modifications and changes may be made to these embodiments without departing from the broader spirit and scope of the invention. Accordingly, the specification and drawings are to be regarded in an illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

1. A method of diverting a packet-switched session, the method including:

receiving a session invitation request propagated from a first party to request establishment of a session with a second party;

detecting a divert trigger event;

responsive to the divert trigger event, originating a fake session initiation request at an intermediate party located intermediate the first and second parties on a network path, the fake session initiation request being constructed to identify the first party as the originator thereof; and

communicating the fake session initiation request to a third party to establish the session between the first and third parties.

2. The method of claim 1 including establishing the session between the first and second parties responsive to the session invitation request prior to communicating the fake session initiation request to the third party; responsive to the divert trigger event, originating a fake session termination request at the intermediate party, the fake termination request being constructed to identify the first party as the originator thereof; and communicating the fake session termination request to the second party to terminate the session between the first and the second parties.

3. The method of claim 1 wherein the divert trigger event comprises un-availability of the second party to receive a call supported by the session.

4. The method of claim 3 wherein the un-availability of the second party is

indicated by queuing of the call at the intermediate party.

5. The method of claim 1 wherein the divert trigger event comprises a transfer of a call supported by the session from the second party to the third party.

6. The method of claim 1 wherein the divert trigger event comprises a request by the first party to place a second call to the third party after the establishment of a first call to the first party, where both the first and second calls are supported by the session.

7. The method of claim 1 wherein the fake session initiation request is constructed to record a network address of the first party as an originating address.

8. The method of claim 1 wherein the fake session initiation request is constructed to record traversal of the fake session initiation request across a network path between the first party and the intermediate party.

9. The method of claim 2 wherein the intermediate party blocks propagation of an acknowledgement, generated by the second party, of the fake session termination request to the first party.

10. The method of claim 1 wherein the intermediate party blocks propagation of an acknowledgement, generated by the third party, of the fake session initiation request to the first party.

11. The method of claim 1 including detecting a revert trigger event; responsive to the revert trigger event, originating a further fake session termination request at the intermediate party, the further fake session termination request identifying the first party as the originator thereof; and

communicating the further fake session termination request to the third party to terminate the session between the first and third parties.

12. The method of claim 11 including originating a further fake session initiation request at the intermediate party, the further fake session initiation request identifying the first party as the originator thereof, and communicating the further fake session initiation request to the second party to re-establish the session between the first party and the second party.

13. The method of claim 11 wherein the revert trigger event comprises the second party becoming available to receive a call supported by the session.

14. The method of claim 11 wherein the revert trigger event comprises queuing of call, supported by the session, between the first and the second parties.

15. The method of claim 11 wherein the revert trigger event comprises termination of a transfer call between the first and the third parties supported by the session.

16. The method of claim 12 wherein the further fake session initiation and termination requests are each constructed to record a network address of the first party as an originating address.

17. The method of claim 12 wherein the further fake session initiation and termination requests are constructed each to record traversal of the respective request across a network path between the first party and the intermediate party.

18. The method of claim 11 wherein the intermediate party blocks propagation of an acknowledgement, generated by the third party, of the

further fake session termination request to the first party.

19. The method of claim 12 wherein the intermediate party blocks propagation of an acknowledgement, generated by the second party, of the further fake session initiation request to the first party.

20. The method of claim 1 wherein the session is established utilizing the Session Initiation Protocol.

21. The method of claim 1 wherein the intermediate party is a proxy agent.

22. The method of claim 1 wherein the session comprises any one of a group including an Internet multi-media conference, an Internet telephone call or a multi-media distribution channel.

23. A server for diverting a packet-switched session, the server being locatable on a network path intermediate first and second parties and including:

a receiver to receive a session invitation request propagated from the first party to request establishment of a session with the second party over a network;

a detector to detect a divert trigger event; and

a protocol module, responsive to the divert trigger event, to originate a fake session initiation request identifying the first party as the originator thereof and to communicate the fake session initiation request to a third party to establish the session between the first and third parties.

24. The server of claim 23 wherein the protocol module establishes the

session between the first and second parties responsive to the session invitation request prior to communication of the fake session initiation request to the third party; responsive to the divert trigger event, originates a fake session termination request, the fake termination request being constructed to identify the first party as the originator thereof, and communicates the fake session termination request to the second party to terminate the session between the first and the second parties.

25. The server of claim 23 wherein the detector detects an un-availability of the second party to receive a call supported by the session as the divert trigger event.

26. The server of claim 25 wherein the detector detects the un-availability of the second party by detecting queuing of the call at the intermediate party.

27. The server of claim 23 wherein the detector detects a transfer of a call supported by the session from the second party to the third party as the divert trigger event.

28. The server of claim 23 wherein the detector detects a request by the first party to place a second call to the third party after the establishment of a first call to the first party as the divert trigger event, where both the first and second calls are supported by the session.

29. The server of claim 23 wherein the protocol module constructs the fake session initiation request to record a network address of the first party as an originating address.

30. The server of claim 23 wherein the protocol module constructs the fake session initiation request to record traversal of the fake session initiation request across a network path between the first party and the server.

31. The server of claim 24 wherein the protocol module is to block propagation of an acknowledgement, generated by the second party, of the fake session termination request to the first party.

32. The server of claim 23 wherein the protocol module is to block propagation of an acknowledgement, generated by the third party, of the fake session initiation request to the first party.

33. The server of claim 23 wherein the detector is to detect a revert trigger event; and the protocol module is, responsive to the revert trigger event, to originate a further fake session termination request, the further fake session termination request identifying the first party as the originator thereof, and to communicate the further fake session termination request to the third party to terminate the session between the first and third parties.

34. The server of claim 33 wherein the protocol module originates a further fake session initiation request, the further fake session initiation request identifying the first party as the originator thereof, and communicates the further fake session initiation request to the second party to re-establish the session between the first party and the second party.

35. The server of claim 33 wherein the detector detects the second party becoming available to receive a call supported by the session as the revert trigger event.

36. The server of claim 33 wherein the detector detects queuing of a call, supported by the session, between the first and the second parties as the revert trigger event.

37. The server of claim 33 wherein the detector detects termination of a

transfer call between the first and the third parties supported by the session as the revert trigger event.

38. The server of claim 34 wherein the protocol module constructs each of the further fake session initiation and termination requests to record a network address of the first party as an originating address.

39. The server of claim 34 wherein the protocol module constructs each of the further fake session initiation and termination requests to record traversal of the respective request across a network path between the first party and the intermediate party.

40. The server of claim 33 wherein the protocol module is to block propagation of an acknowledgement, generated by the third party, of the further fake session termination request to the first party.

41. The server of claim 24 wherein the protocol module is to block propagation of an acknowledgement, generated by the second party, of the further fake session initiation request to the first party.

42. The server of claim 23 wherein the session is established utilizing the Session Initiation Protocol.

43. The server of claim 23 wherein the server functions as a proxy agent.

44. The server of claim 23 wherein the session comprises any one of a group including an Internet multi-media conference, an Internet telephone call or a multi-media distribution channel.

45. A server for diverting a packet-switched session, the server being locatable on a network path intermediate first and second parties and

including:

first means for receiving a session invitation request propagated from the first party to request establishment of a session with the second party over a network;

second means for detecting a divert trigger event; and

third means, responsive to the divert trigger event, for originating a fake session initiation request identifying the first party as the originator thereof and for communicating the fake session initiation request to a third party to establish the session between the first and third parties.

46. A machine-readable medium storing a sequence of instructions that, when executed by a machine, cause the machine to divert a packet session by:

receiving a session invitation request propagated from a first party to request establishment of a session with a second party;

detecting a divert trigger event;

responsive to the divert trigger event, originating a fake session initiation request at an intermediate party located intermediate the first and second parties on a network path, the fake session initiation request being constructed to identify the first party as the originator thereof; and

communicating the fake session initiation request to a third party to establish the session between the first and third parties.

47. The machine-readable medium of claim 46 wherein the sequence of instructions causes the machine to establish the session between the first and

second parties responsive to the session invitation request prior to communicating the fake session initiation request to the third party; responsive to the divert trigger event, to originate a fake session termination request at the intermediate party, the fake termination request being constructed to identify the first party as the originator thereof; and to communicate the fake session termination request to the second party to terminate the session between the first and the second parties.

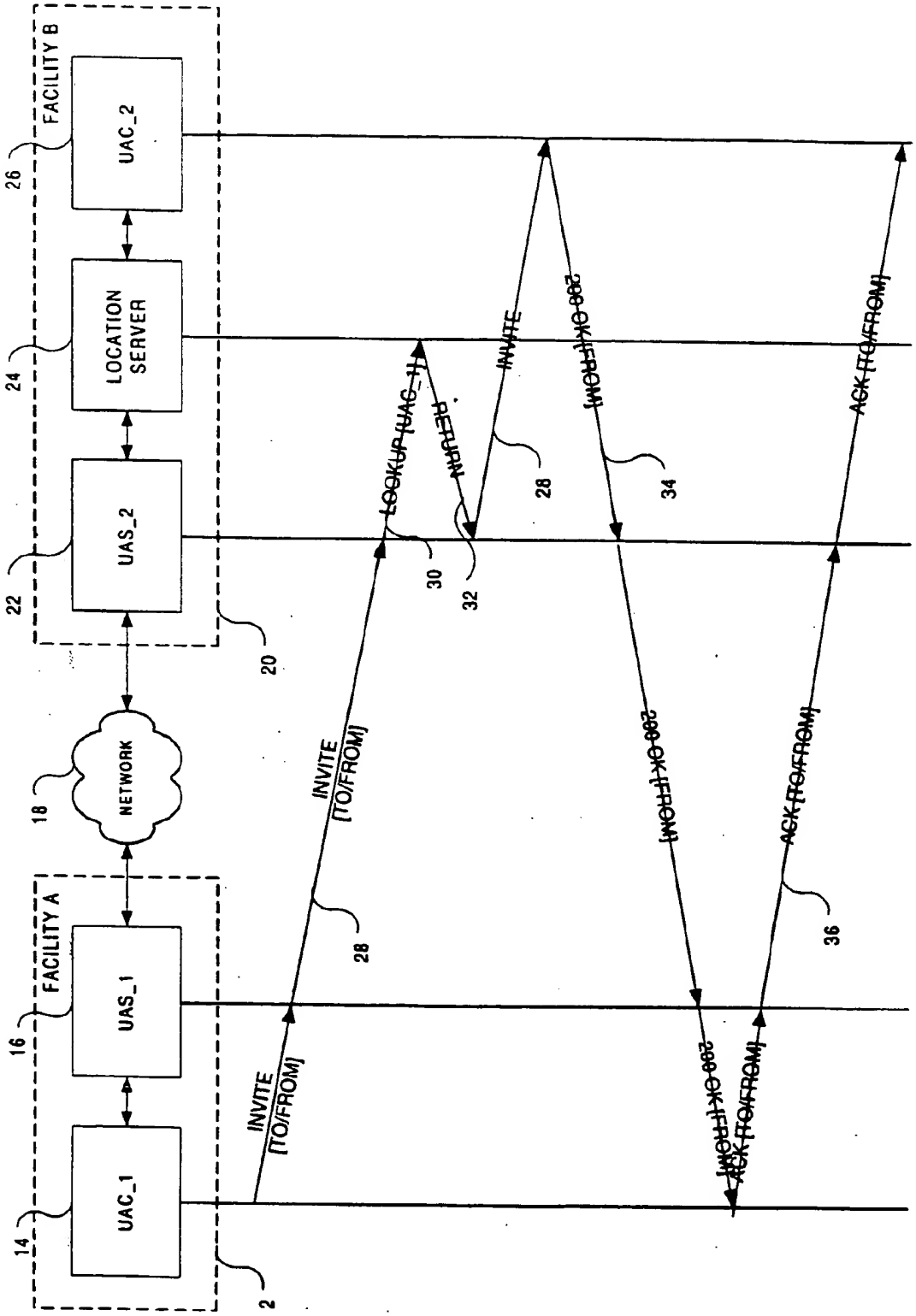

FIGURE 1A
(PRIOR ART)

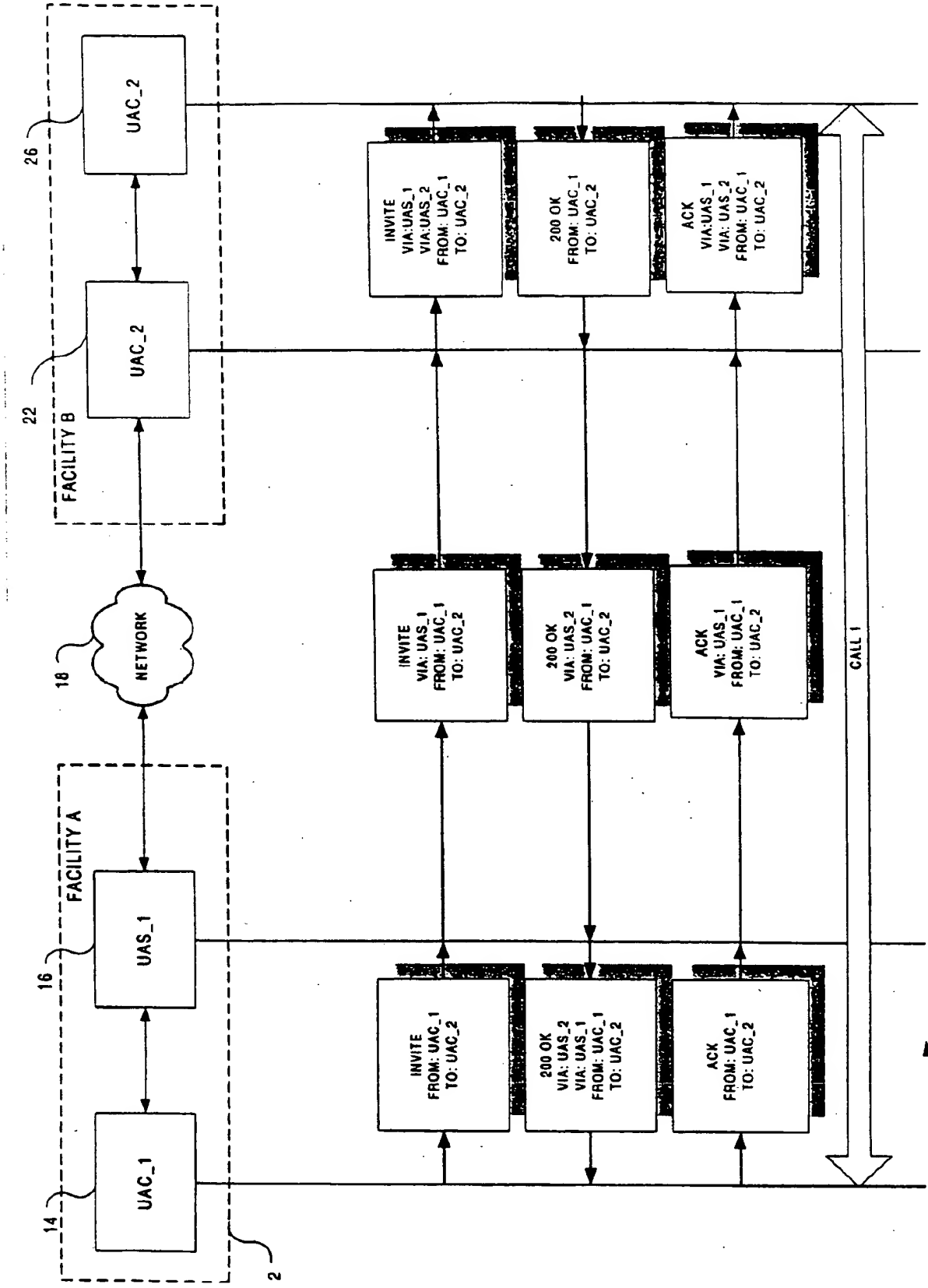

FIGURE 1B
(PRIOR ART)

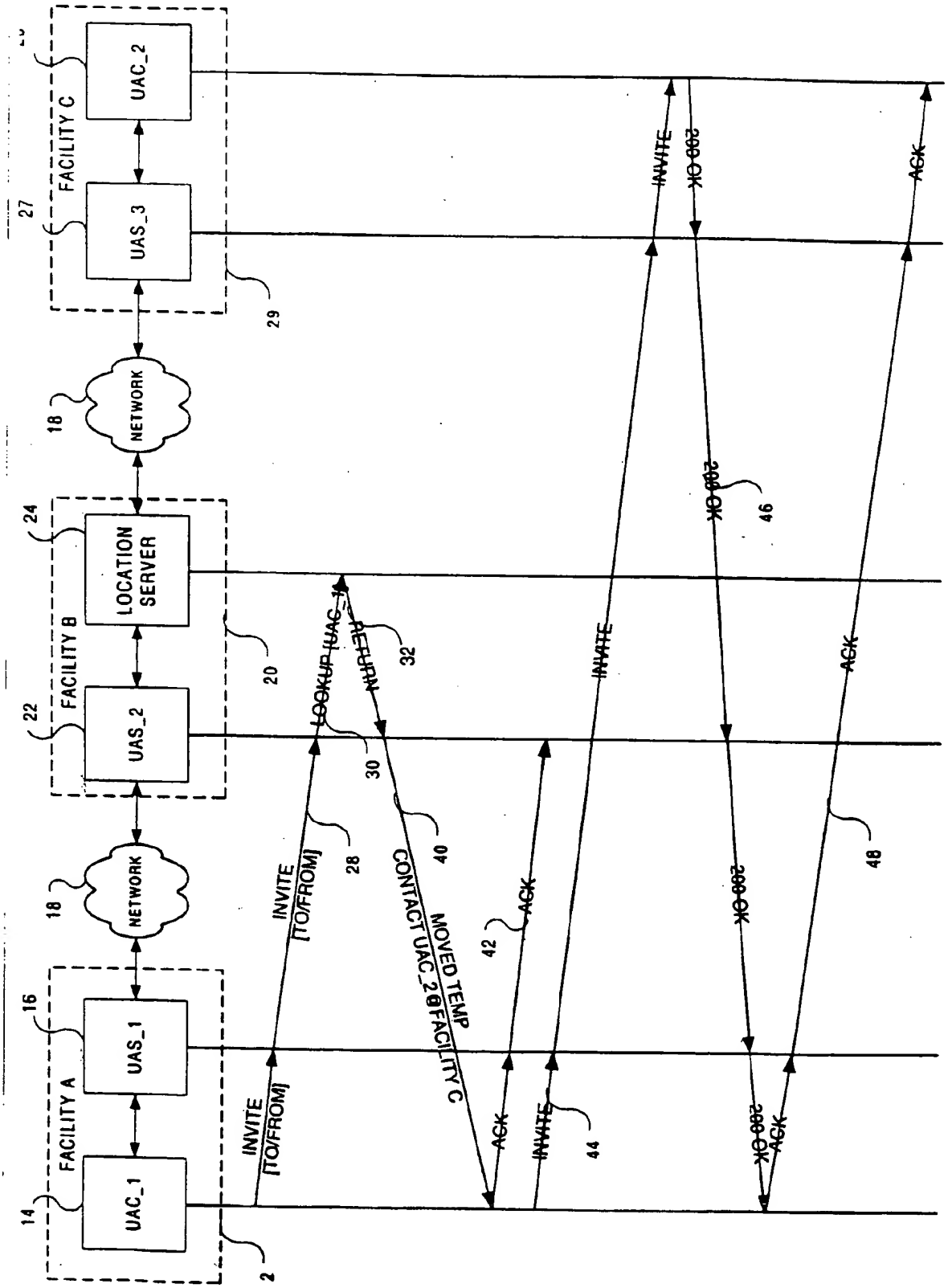

FIGURE 2
(PRIOR ART)

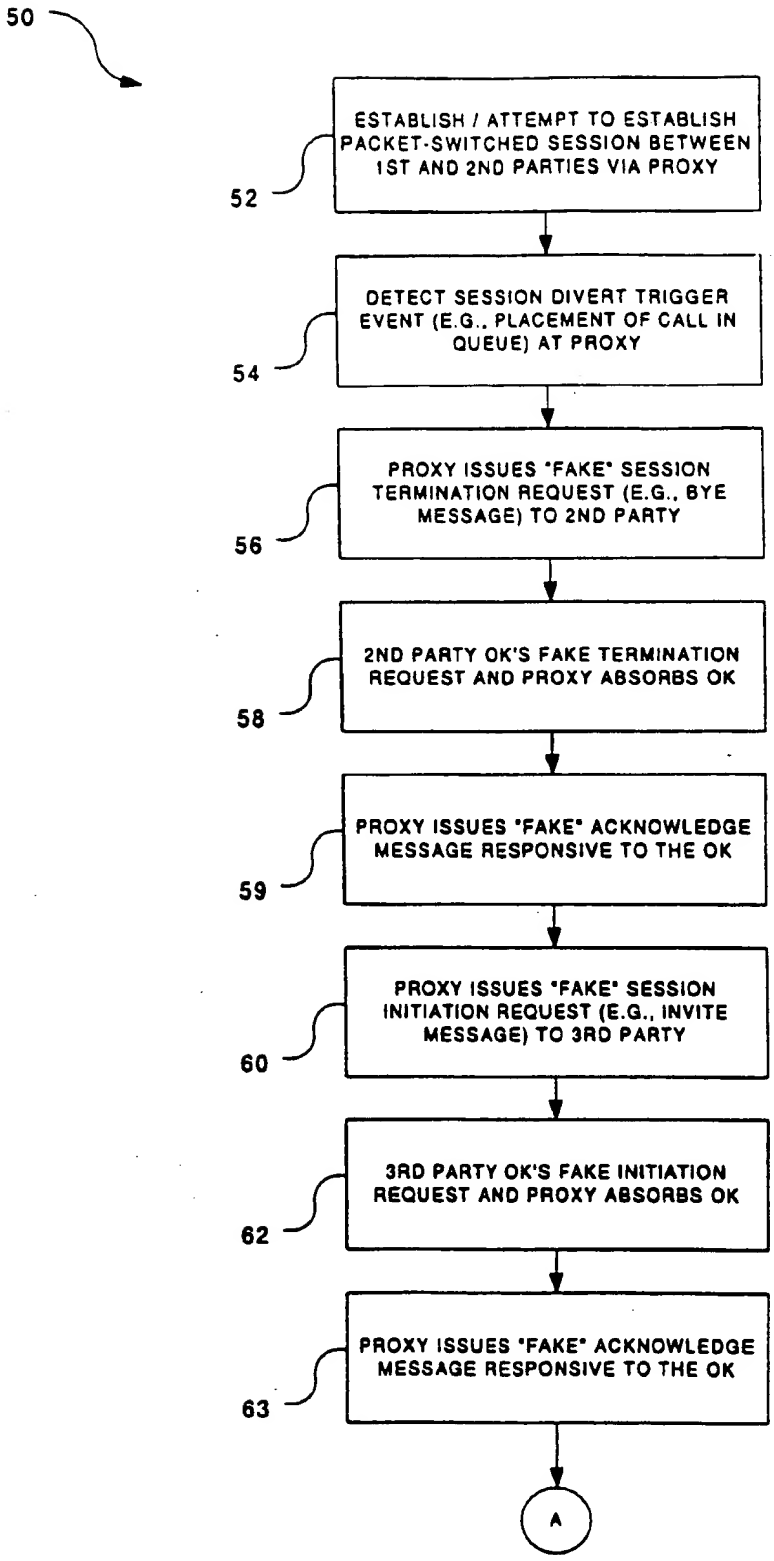
50
ESTABLISH / ATTEMPT TO ESTABLISH PACKET-SWITCHED SESSION BETWEEN 1ST AND 2ND PARTIES VIA PROXY
52
DETECT SESSION DIVERT TRIGGER EVENT (E.G., PLACEMENT OF CALL IN QUEUE) AT PROXY
54
PROXY ISSUES "FAKE" SESSION TERMINATION REQUEST (E.G., BYE MESSAGE) TO 2ND PARTY
56
2ND PARTY OK'S FAKE TERMINATION REQUEST AND PROXY ABSORBS OK
58
PROXY ISSUES "FAKE" ACKNOWLEDGE MESSAGE RESPONSIVE TO THE OK
59
PROXY ISSUES "FAKE" SESSION INITIATION REQUEST (E.G., INVITE MESSAGE) TO 3RD PARTY
60
3RD PARTY OK'S FAKE INITIATION REQUEST AND PROXY ABSORBS OK
62
PROXY ISSUES "FAKE" ACKNOWLEDGE MESSAGE RESPONSIVE TO THE OK
63
A

FIGURE 3

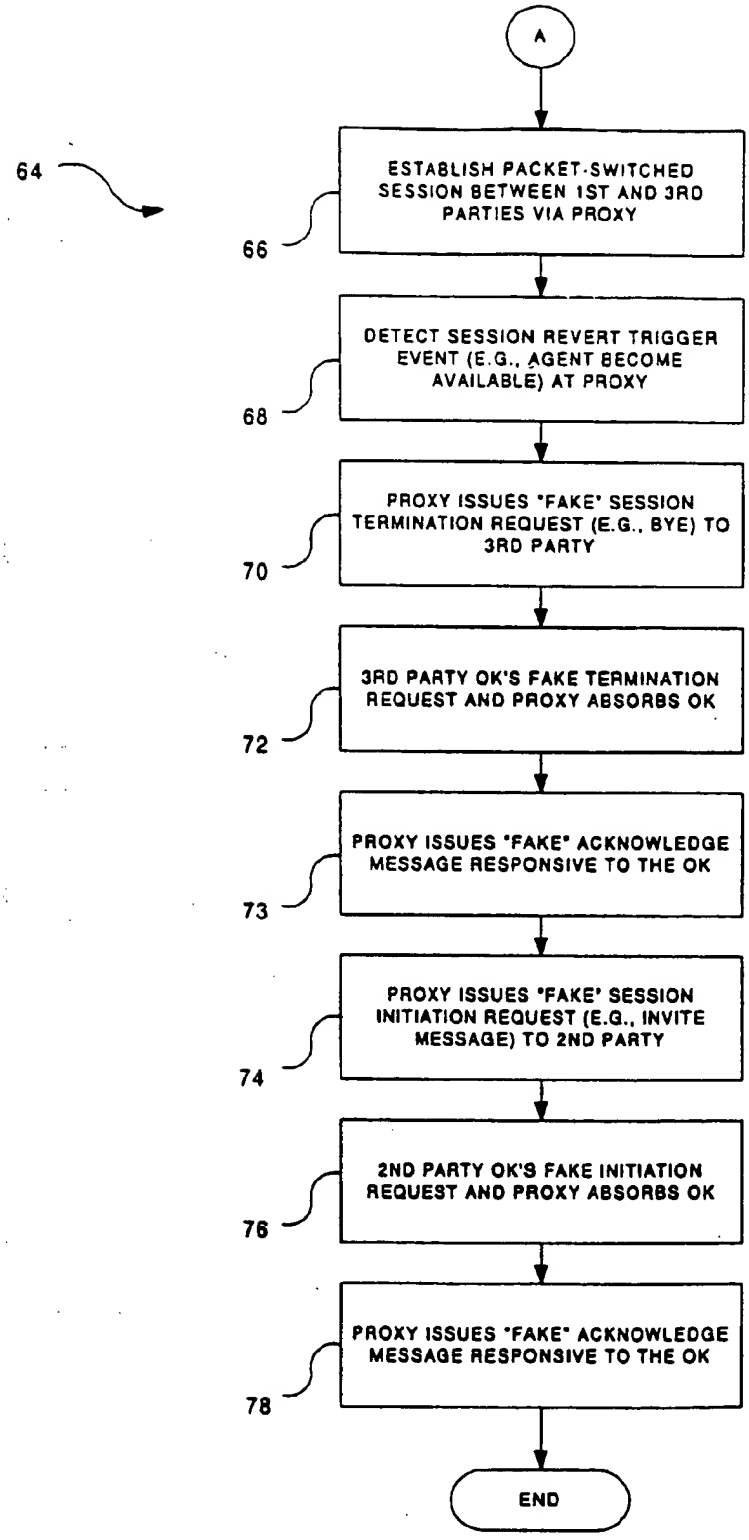
A
64
ESTABLISH PACKET-SWITCHED SESSION BETWEEN 1ST AND 3RD PARTIES VIA PROXY
66
DETECT SESSION REVERT TRIGGER EVENT (E.G., AGENT BECOME AVAILABLE) AT PROXY
68
PROXY ISSUES "FAKE" SESSION TERMINATION REQUEST (E.G., BYE) TO 3RD PARTY
70
3RD PARTY OK'S FAKE TERMINATION REQUEST AND PROXY ABSORBS OK
72
PROXY ISSUES "FAKE" ACKNOWLEDGE MESSAGE RESPONSIVE TO THE OK
73
PROXY ISSUES "FAKE" SESSION INITIATION REQUEST (E.G., INVITE MESSAGE) TO 2ND PARTY
74
2ND PARTY OK'S FAKE INITIATION REQUEST AND PROXY ABSORBS OK
76
PROXY ISSUES "FAKE" ACKNOWLEDGE MESSAGE RESPONSIVE TO THE OK
78
END

FIGURE 4

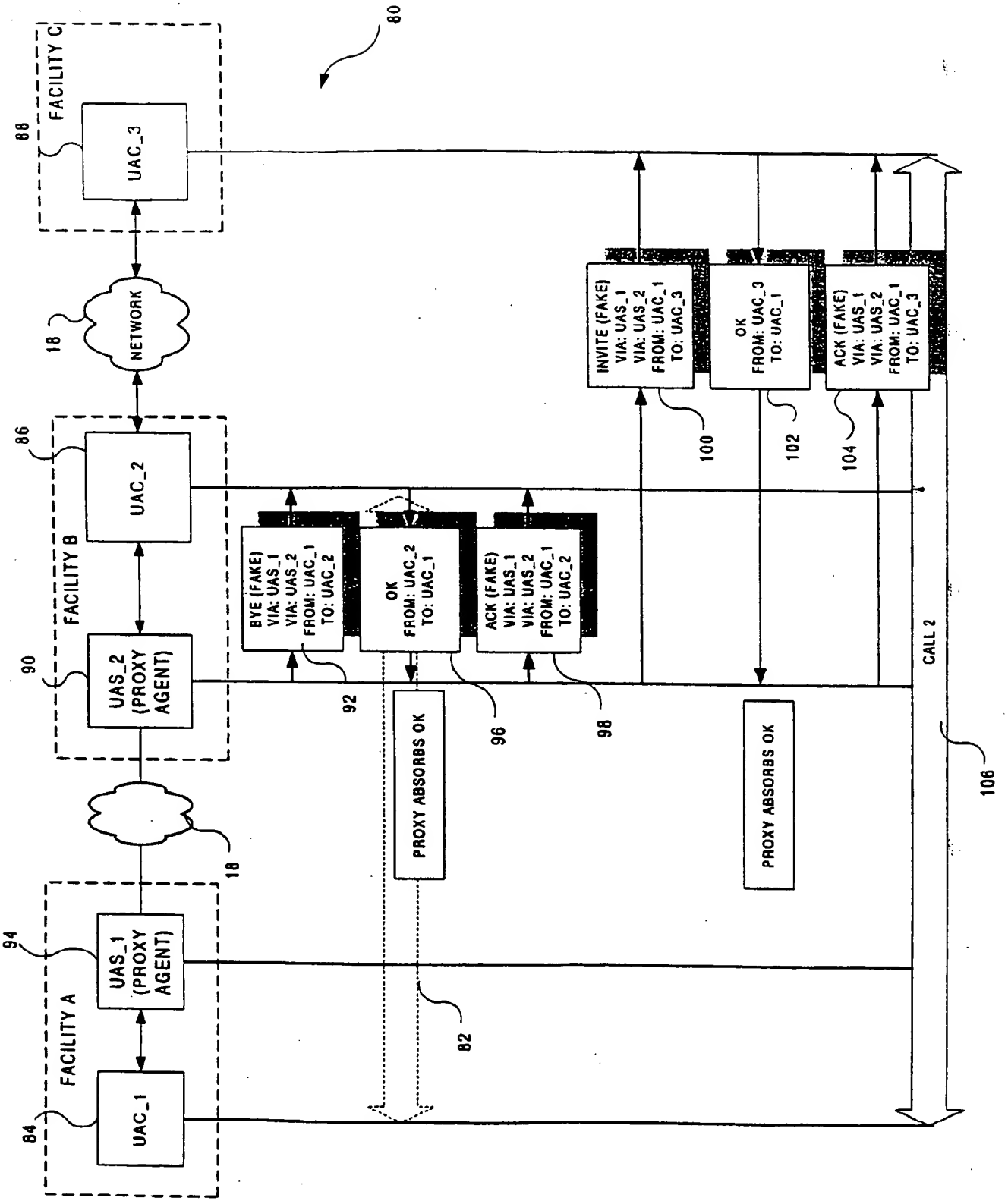
80
FACILITY C
88
UAC_3
NETWORK
18
FACILITY B
86
UAC_2
90
UAS_2 (PROXY AGENT)
18
FACILITY A
94
UAS_1 (PROXY AGENT)
84
UAC_1
82
INVITE (FAKE) VIA: UAS_1 VIA: UAS_2 FROM: UAC_1 TO: UAC_3
OK FROM: UAC_3 TO: UAC_1
ACK (FAKE) VIA: UAS_1 VIA: UAS_2 FROM: UAC_1 TO: UAC_3
100
102
104
BYE (FAKE) VIA: UAS_1 VIA: UAS_2 FROM: UAC_1 TO: UAC_2
OK FROM: UAC_2 TO: UAC_1
ACK (FAKE) VIA: UAS_1 VIA: UAS_2 FROM: UAC_1 TO: UAC_2
92
96
98
PROXY ABSORBS OK
PROXY ABSORBS OK
CALL 2
106

FIGURE 5A

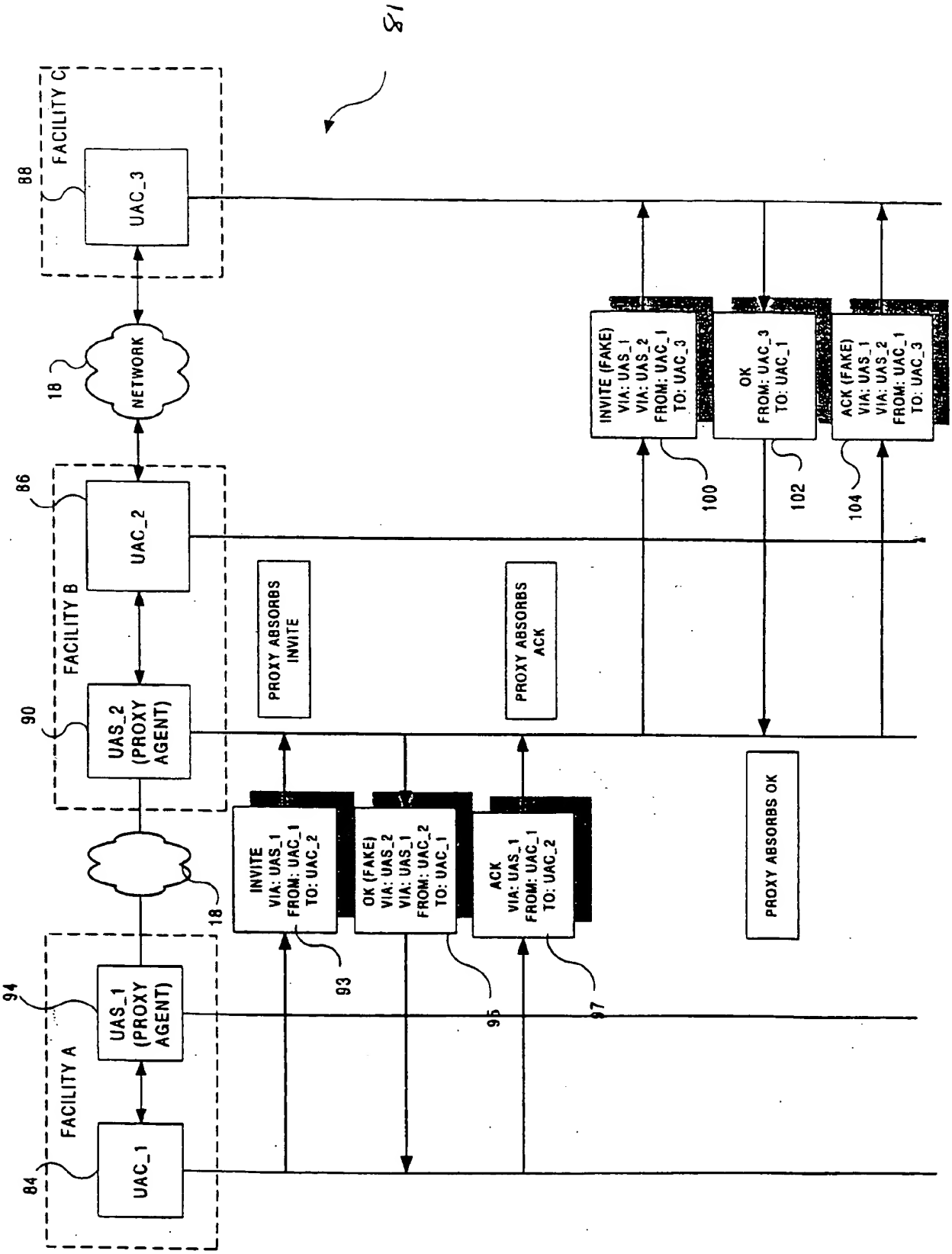
81
FACILITY C
88
UAC_3
NETWORK
18
FACILITY B
86
UAC_2
90
UAS_2 (PROXY AGENT)
PROXY ABSORBS INVITE
PROXY ABSORBS ACK
INVITE (FAKE) VIA: UAS_1 VIA: UAS_2 FROM: UAC_1 TO: UAC_3
OK FROM: UAC_3 TO: UAC_1
ACK (FAKE) VIA: UAS_1 VIA: UAS_2 FROM: UAC_1 TO: UAC_3
100
102
104
INVITE VIA: UAS_1 FROM: UAC_1 TO: UAC_2
OK (FAKE) VIA: UAS_2 VIA: UAS_1 FROM: UAC_2 TO: UAC_1
ACK VIA: UAS_1 FROM: UAC_1 TO: UAC_2
PROXY ABSORBS OK
93
95
97
94
UAS_1 (PROXY AGENT)
FACILITY A
84
UAC_1

FIGURE 5B

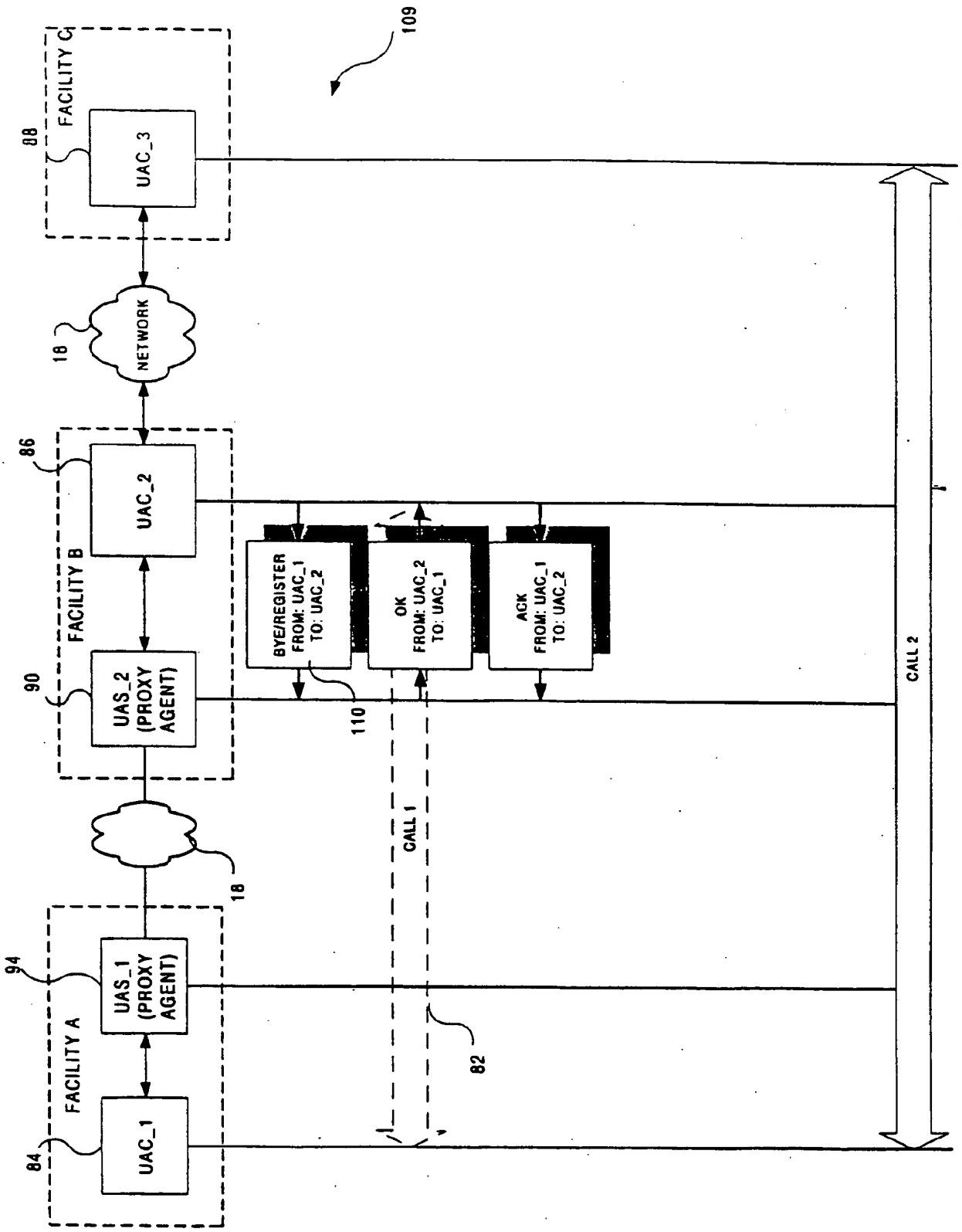
109
FACILITY C
88
UAC_3
NETWORK
18
86
FACILITY B
UAC_2
90
UAS_2 (PROXY AGENT)
BYE/REGISTER FROM: UAC_1 TO: UAC_2
OK FROM: UAC_2 TO: UAC_1
ACK FROM: UAC_1 TO: UAC_2
110
CALL 1
CALL 2
18
94
FACILITY A
UAS_1 (PROXY AGENT)
82
84
UAC_1

FIGURE 6

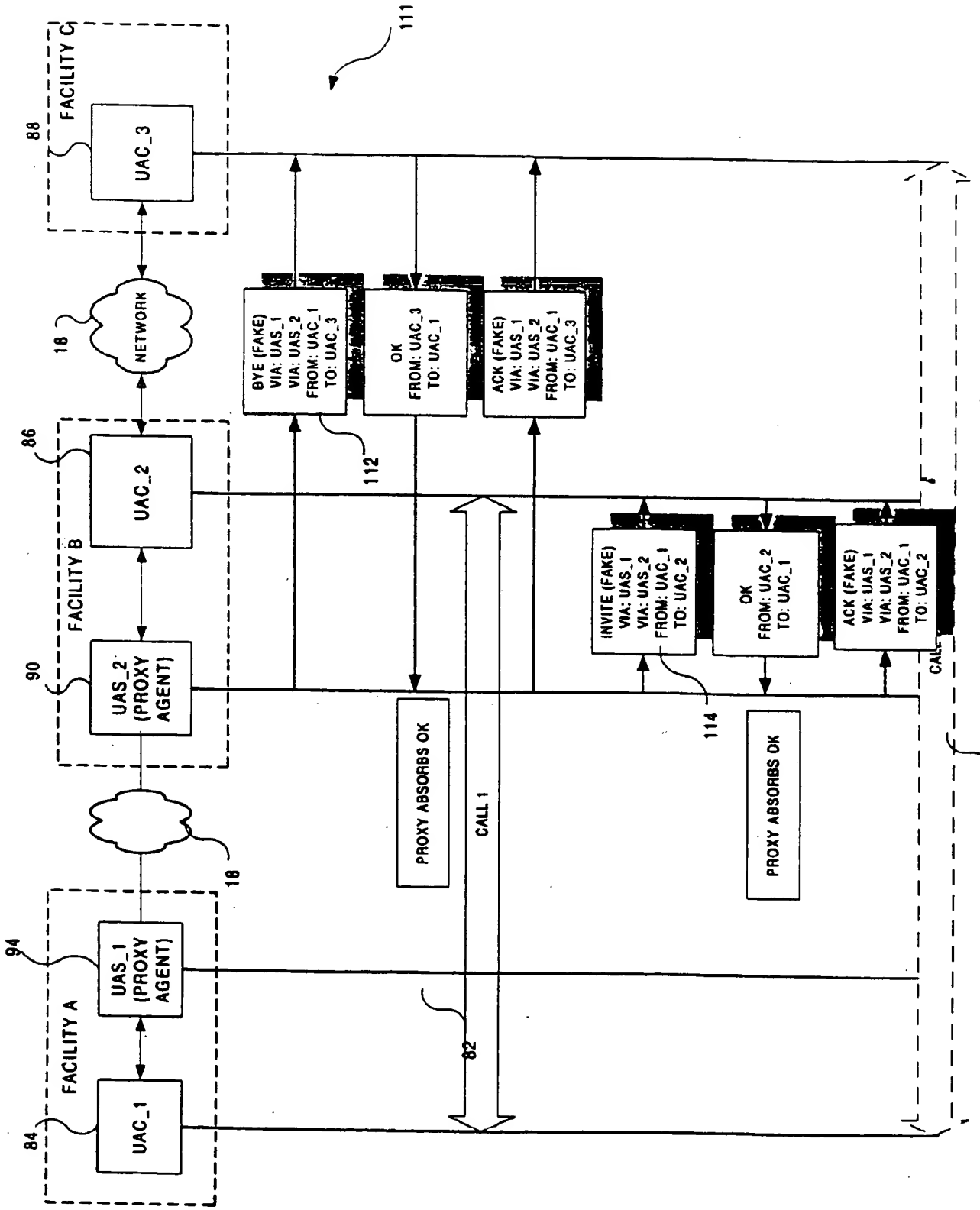
111
FACILITY C
88
UAC_3
NETWORK
18
BYE (FAKE)
VIA: UAS_1
VIA: UAS_2
FROM: UAC_1
TO: UAC_3
OK
FROM: UAC_3
TO: UAC_1
ACK (FAKE)
VIA: UAS_1
VIA: UAS_2
FROM: UAC_1
TO: UAC_3
112
86
UAC_2
FACILITY B
INVITE (FAKE)
VIA: UAS_1
VIA: UAS_2
FROM: UAC_1
TO: UAC_2
OK
FROM: UAC_2
TO: UAC_1
ACK (FAKE)
VIA: UAS_1
VIA: UAS_2
FROM: UAC_1
TO: UAC_2
114
90
UAS_2
(PROXY
AGENT)
PROXY ABSORBS OK
PROXY ABSORBS OK
CALL 1
82
106
18
94
UAS_1
(PROXY
AGENT)
FACILITY A
84
UAC_1

FIGURE 7

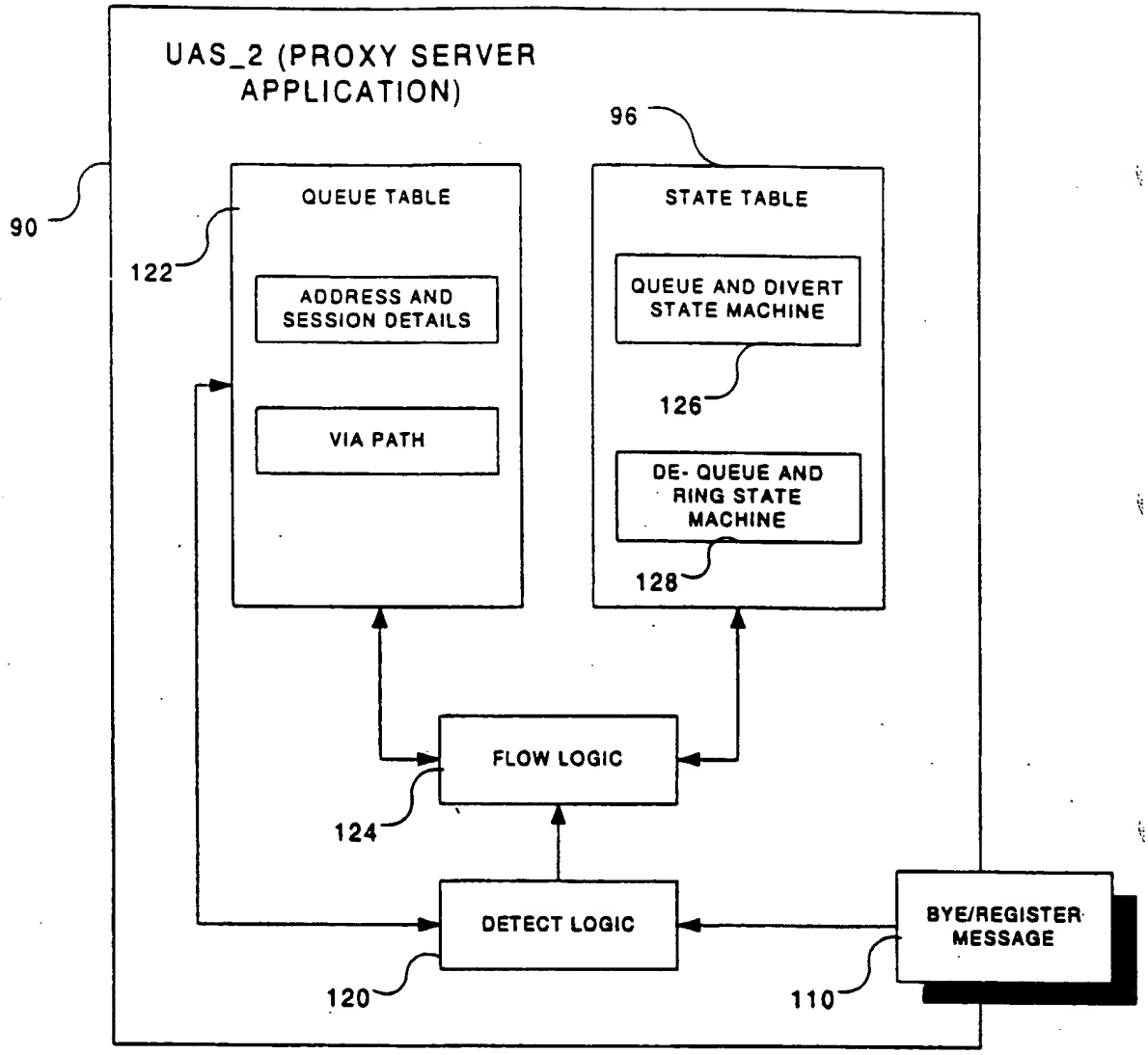
UAS_2 (PROXY SERVER APPLICATION)
90
96
QUEUE TABLE
122
ADDRESS AND SESSION DETAILS
VIA PATH
STATE TABLE
QUEUE AND DIVERT STATE MACHINE
126
DE- QUEUE AND RING STATE MACHINE
128
FLOW LOGIC
124
DETECT LOGIC
120
BYE/REGISTER MESSAGE
110

FIGURE 8

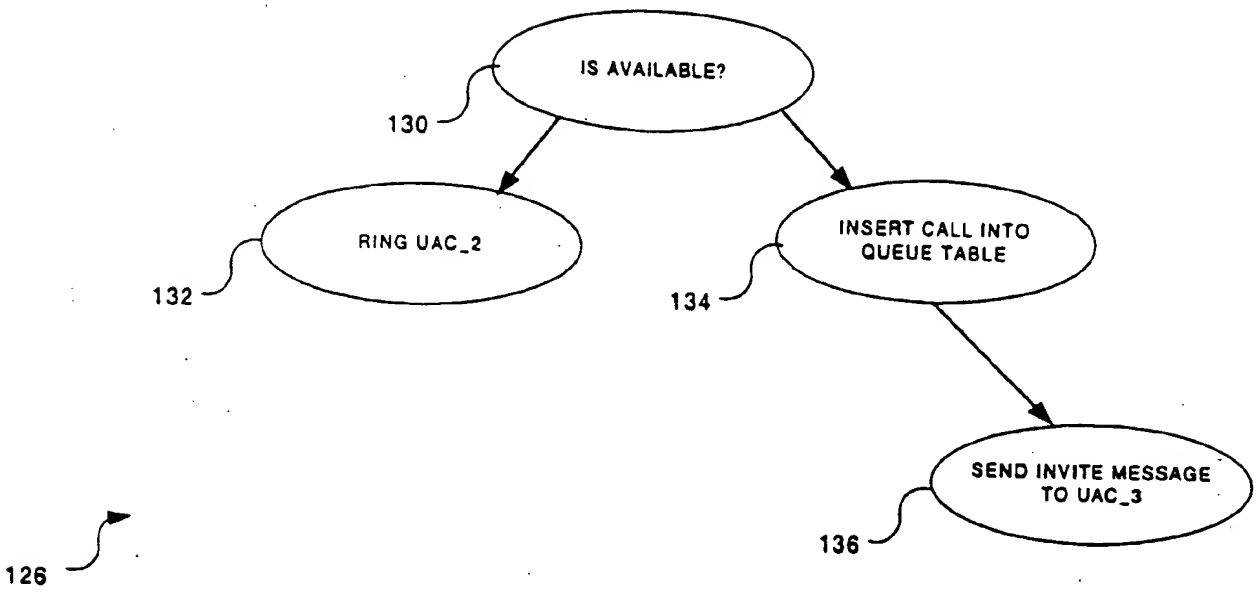

FIGURE 9

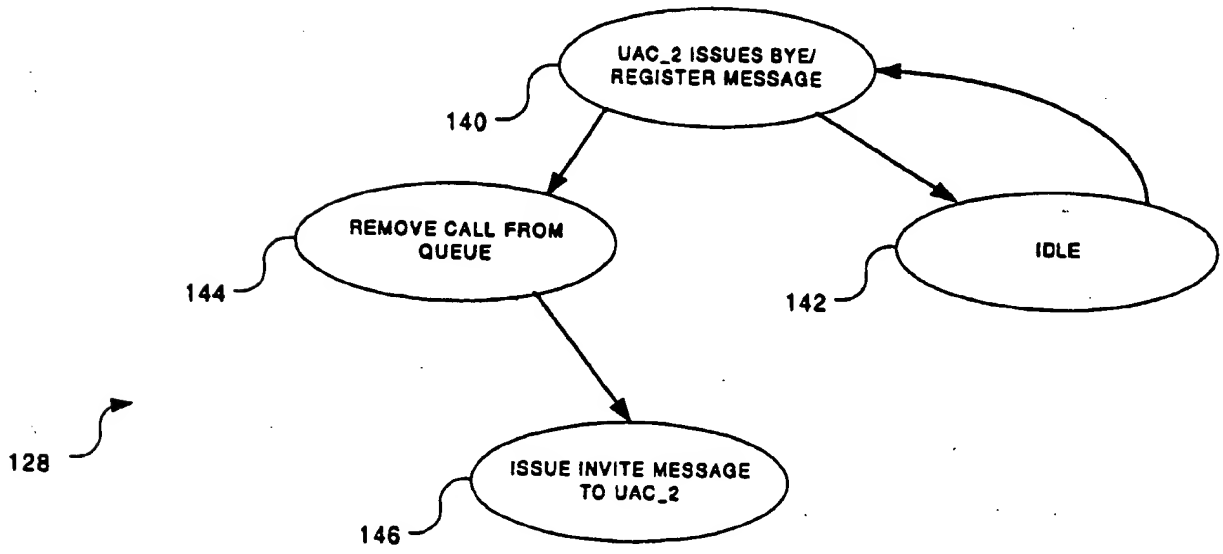

FIGURE 10

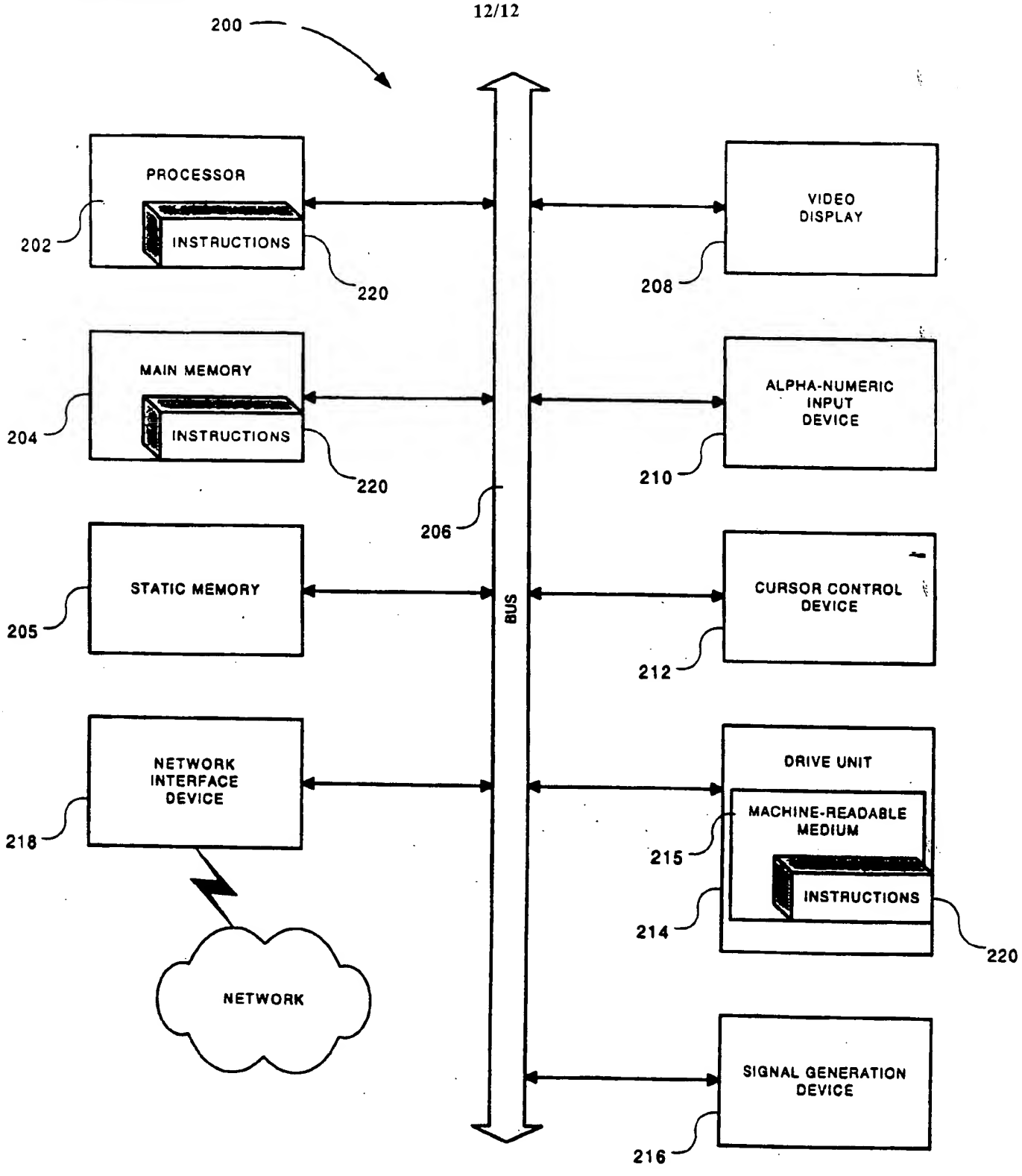
200
PROCESSOR
INSTRUCTIONS
202
220
MAIN MEMORY
INSTRUCTIONS
204
220
STATIC MEMORY
205
NETWORK INTERFACE DEVICE
218
NETWORK
206
BUS
VIDEO DISPLAY
208
ALPHA-NUMERIC INPUT DEVICE
210
CURSOR CONTROL DEVICE
212
DRIVE UNIT
MACHINE-READABLE MEDIUM
215
INSTRUCTIONS
214
220
SIGNAL GENERATION DEVICE
216

FIGURE 11